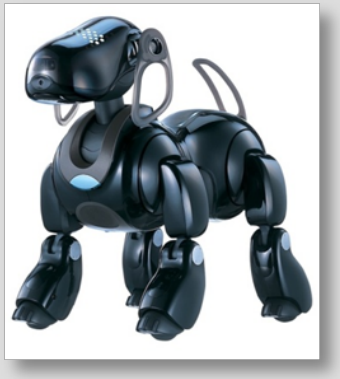
Image transformations



Prof. Noah Snavely
CS1114
http://cs1114.cs.cornell.edu

Cornell University
Computer Science

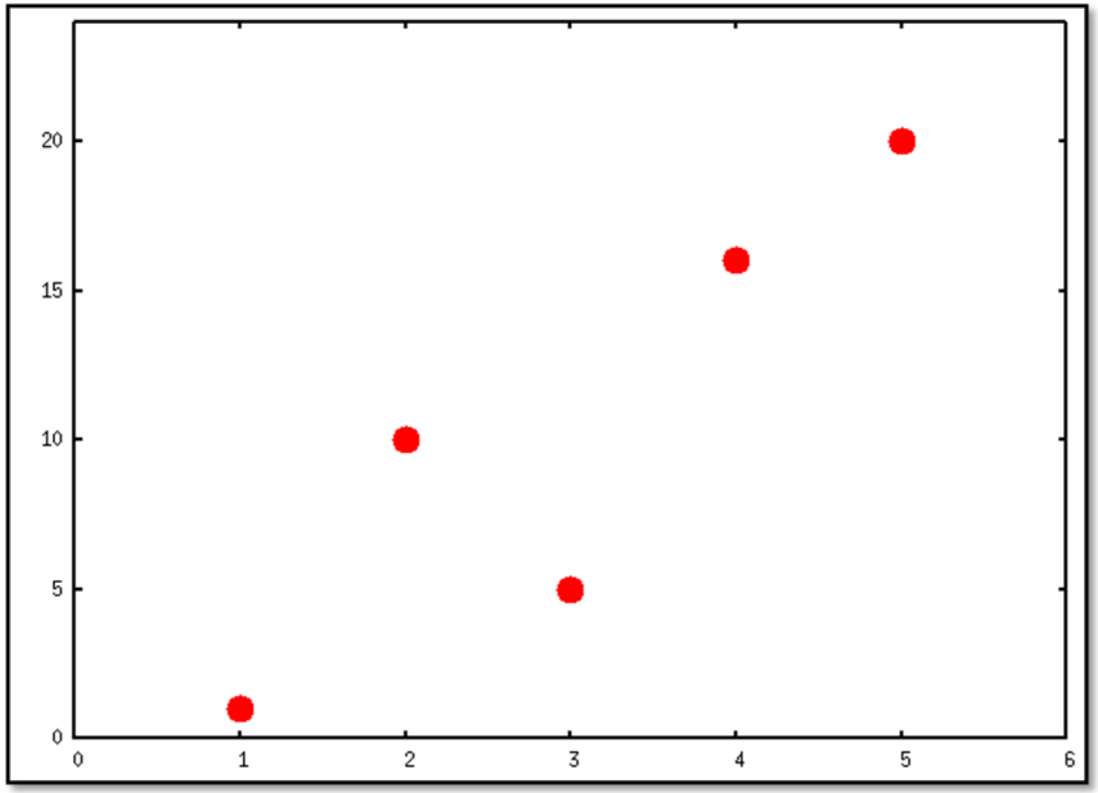
Administrivia

- Assignment 4 will be out by tomorrow
  - Due the Friday after spring break
- Quiz 3 next time
  - Topics: convex hull, interpolation, image transformations

Last time: interpolation

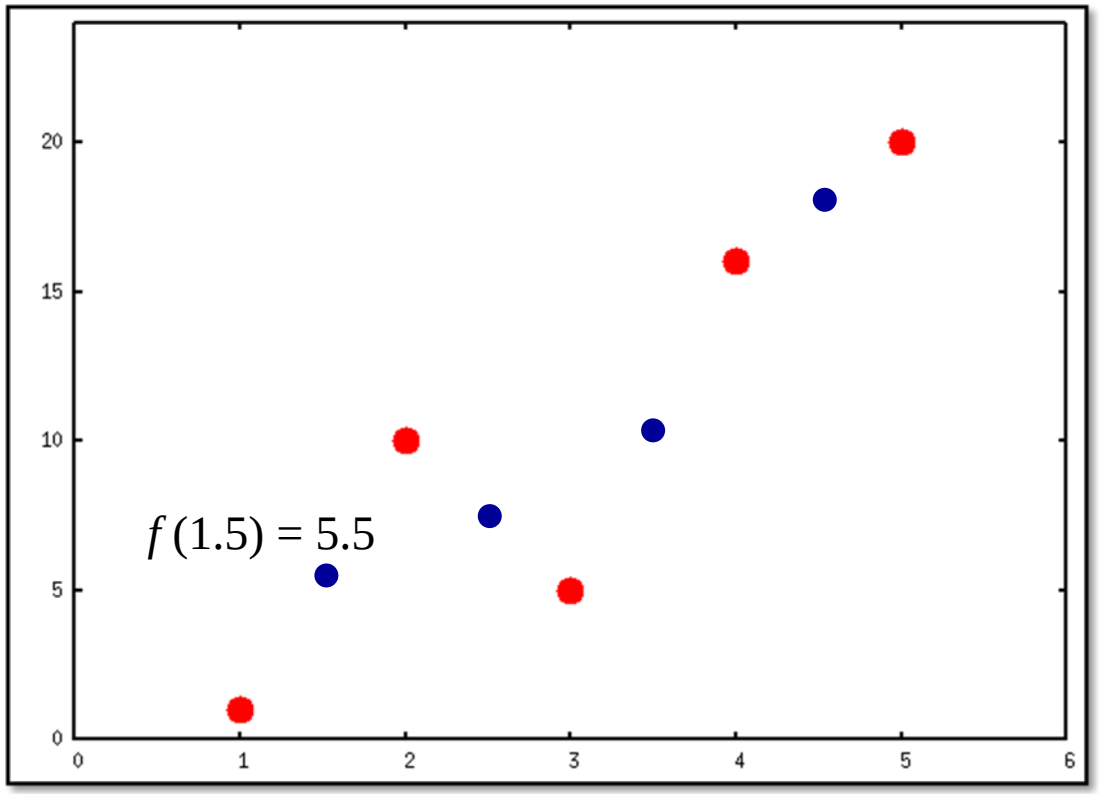
- Example:

  f(1) = 1, f(2) = 10, f(3) = 5 , f(4) = 16, f(5) = 20



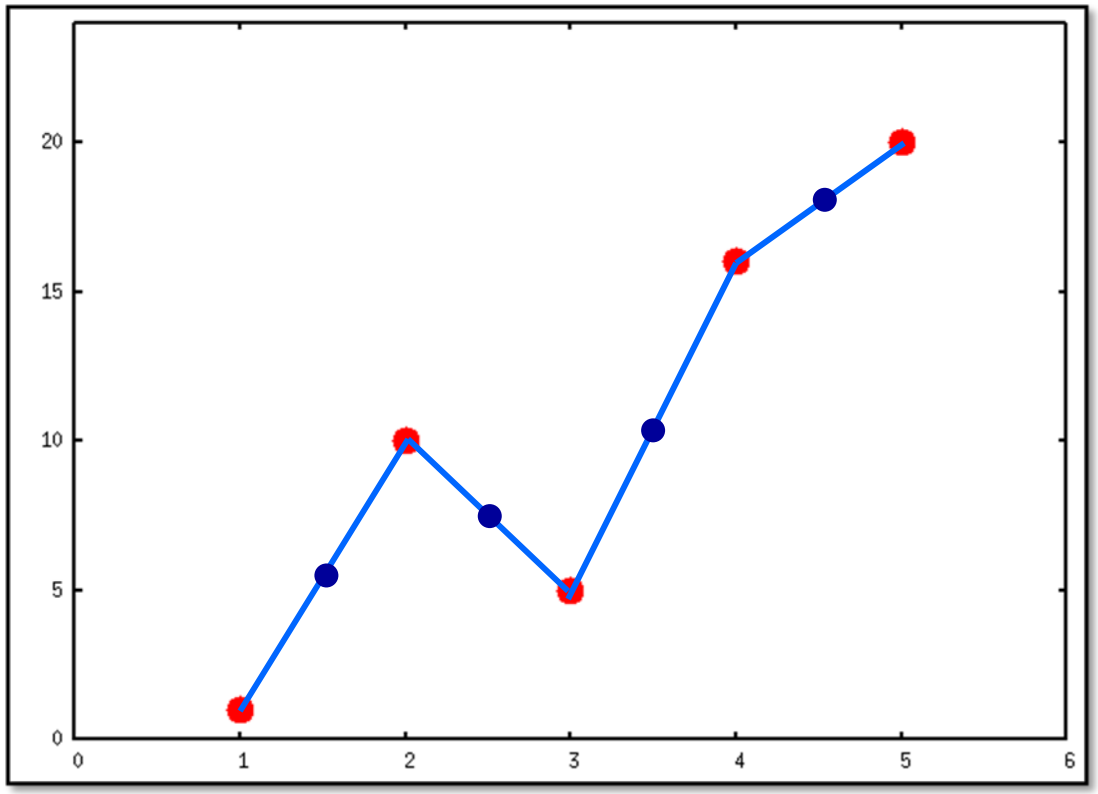
Interpolation

- How can we find f(1.5)?

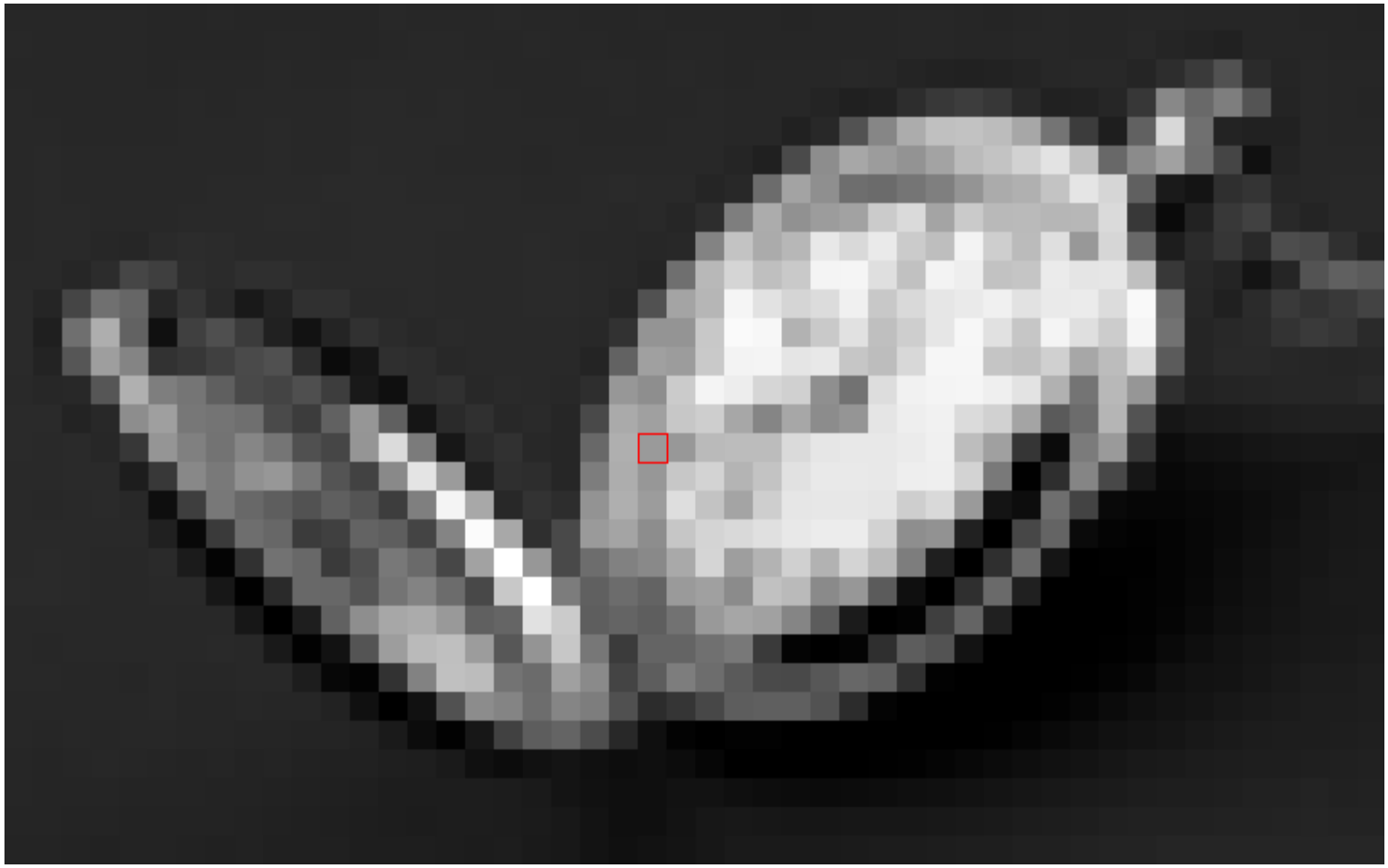


Linear interpolation (lerp)

- Fit a line between each pair of data points



Nearest neighbor interpolation



Bilinear interpolation

Bicubic interpolation

Today: image transformations

2D Transformations

- 2D Transformation:
  - Function from 2D → 2D

$$f(x, y) = (x', y')$$

  - We'll apply this function to every pixel to get a new pixel location

- Examples:

$$f(x, y) = (0.5x, 1.5y)$$

$$f(x, y) = (y, x)$$

2D Transformations

$f(x, y) = (0.5x, 2y)$

2D Transformations

$f(x, y) = (y, x)$

2D Transformations

- Can be non-linear:

2D Transformations

image credit: Matt Brown

Linear transformations

- We will focus on linear transformations
- 1D:

$$f(x) = ax$$

- 2D:

$$f(x, y) = (ax + by, cx + dy)$$

- Examples

1. $f(x, y) = (0.5x, 1.5y)$
2. $f(x, y) = (y, x)$

2D Linear Transformations

- Can be represented with a 2D matrix

$$T = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

- And applied to a point using matrix multiplication

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} ax + by \\ cx + dy \end{bmatrix}$$

2D Linear Transformations

- Can be represented with a 2D matrix

$$T = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \qquad p = \begin{bmatrix} x \\ y \end{bmatrix}$$

$$Tp = q$$

Examples

- f(x, y) = (0.5x, 1.5y)

$$T = \begin{bmatrix} 0.5 & 0 \\ 0 & 1.5 \end{bmatrix}$$

- f(x, y) = (y, x)

$$T = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Common linear transformations

- Uniform scaling:

$$S = \begin{bmatrix} s & 0 \\ 0 & s \end{bmatrix}$$

$$\begin{bmatrix} s & 0 \\ 0 & s \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} sx \\ sy \end{bmatrix}$$

Common linear transformations

- Rotation by angle θ

$$R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

Common linear transformations

- Shear

$$H = \begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix}$$

Composing linear transformations

- What if we want to scale *and* rotate?
- Answer: multiply the matrices together

$$S = \begin{bmatrix} s & 0 \\ 0 & s \end{bmatrix} \qquad R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

scale *rotation*

$$RS = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} s & 0 \\ 0 & s \end{bmatrix} = \begin{bmatrix} s\cos\theta & -s\sin\theta \\ s\sin\theta & s\cos\theta \end{bmatrix}$$

scale and rotation

Implementing transformations

- First approach (grayscale image)

```
function img_out = transform_image(img_in, T)

[num_rows, num_cols] = size(img_in);
img_out = zeros(num_rows, num_cols);

for row = 1:num_rows
    for col = 1:num_cols
        p = [col; row];
        p_new = T * p;
        img_out(p_new(2), p_new(1)) = img_in(row, col);
    end
end
```

To Matlab

Forward mapping

- Lots of problems came up…

How do we fix this?

- Answer: do the opposite

1. Create an output image
2. For each pixel in the output image, find the corresponding pixel in the input image
3. Give that output pixel the same color

- Requires that we invert the mapping

Inverse mapping

- How do we invert the mapping?

- With linear transformations T, we invert T

$$T = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \quad T^{-1}T = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$T^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

Inverse mapping

Resampling

- Suppose we scale image by 2
- T = [2 0 ; 0 2]

- inv(T) =

- Pixel (5,5) in img_out should be colored with pixel (2.5, 2.5) in img_in

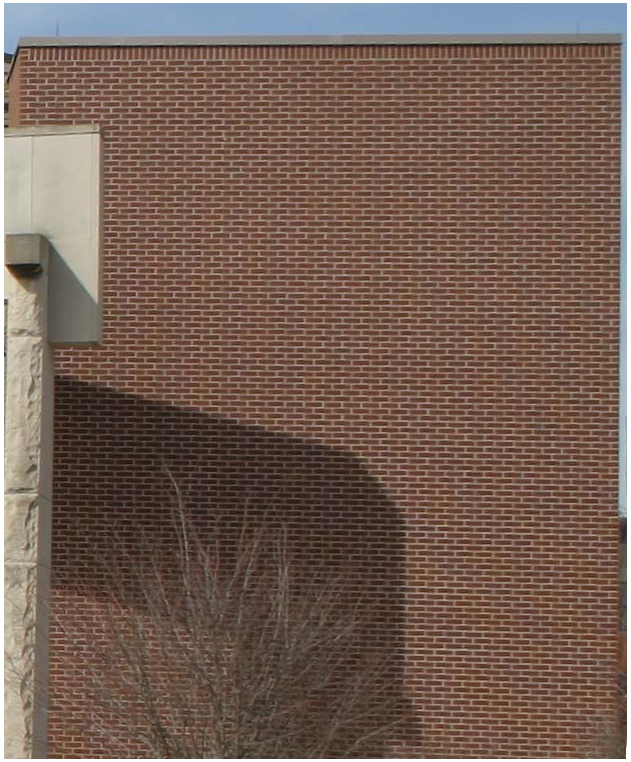
- How do we find the intensity at (2.5, 2.5)?

Inverse mapping

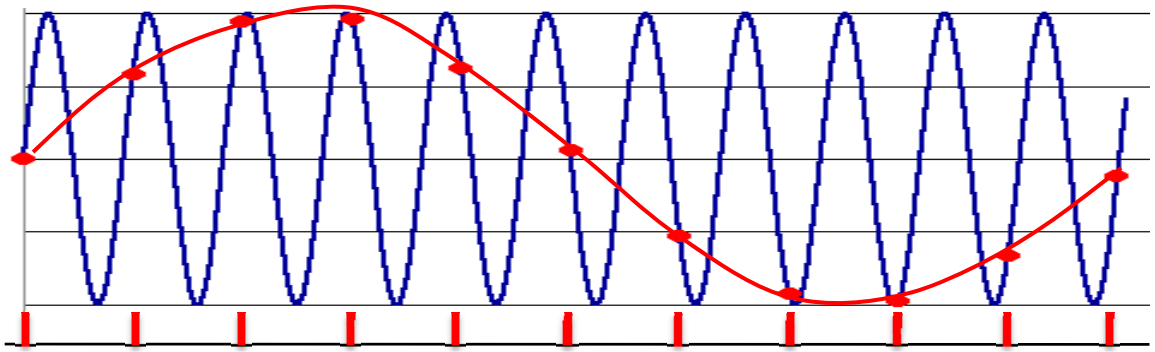
Downsampling

- Suppose we scale image by 0.25

Downsampling



What's going on?



- **Aliasing** can arise when you sample a continuous signal or image
- Occurs when the sampling rate is not high enough to capture the detail in the image
- Can give you the wrong signal/image—an alias